INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO
Ingeniería en Sistemas Computacionales

Compiladores

Práctica 3 | Algoritmo de Subconjuntos

Flores Alvarado Diego Armando – Boleta: 2015340065
Profesor **|** Saucedo Delgado Rafael Norman
3CV6

Ciudad de México, lunes, 9 de noviembre de 2020

Introducción

Autómata Finito Determinista

Un autómata finito determinista (AFD) consta de:

1. Un conjunto finito de estados, a menudo designado como Q.
2. Un conjunto finito de símbolos de entrada, a menudo designado como Σ.
3. Una función de transición que toma como argumentos un estado y un símbolo de entrada y devuelve un estado. La función de transición se designa habitualmente como δ. En nuestra representación gráfica informal del autómata, δ se ha representa mediante arcos entre los estados y las etiquetas sobre los arcos. Si q es un estado y a es un símbolo de entrada, entonces δ(q,a) es el estado p tal que existe un arco etiquetado a que va desde q hasta p^2.
4. Un estado inicial, uno de los estados de Q.
5. Un conjunto de estados finales o de aceptación F. El conjunto F es un subconjunto de Q

Autómata Finito No Determinista

Un autómata finito “no determinista” (AFN) tiene la capacidad de estar en varios estados a la vez. Esta capacidad a menudo se expresa como la posibilidad de que el autómata “conjeture” algo acerca de su entrada. Por ejemplo, cuando el autómata se utiliza para buscar determinadas secuencias de caracteres (por ejemplo, palabras clave) dentro de una cadena de texto larga, resulta útil “conjeturar” que estamos al principio de una de estas cadenas y utilizar una secuencia de estados únicamente para comprobar la aparición de la cadena, carácter por carácter

Algoritmo de los Subconjuntos

La idea general de la construcción de subconjuntos es que cada estado del AFD construido corresponde a un conjunto de estados del AFN. Después de leer la entrada $a_1 a_2 \cdots a_n$, el AFD se encuentra en el estado que corresponde al conjunto de estados que el AFN puede alcanzar, desde su estado inicial, siguiendo los caminos etiquetados como $a_1 a_2 \cdots a_n$. Para poder realizar una conversión de un AFN a un AFD se utiliza el siguiente algoritmo:

Entrada:	Un AFN ***N.***
Salida:	Un AFD que acepta el mismo lenguaje que *N*.
Método:	Se construye una tabla de transición *Dtran* para D. Cada estado de D es un conjunto de estados del AFN. Para poder manejar las transiciones ***ε*** se apoyan de las operaciones mostradas en la Ilustracion C3 – 1.

OPERACIÓN	DESCRIPCIÓN
ϵ-*cerradura*(s)	Conjunto de estados del AFN a los que se puede llegar desde el estado s del AFN, sólo en las transiciones ϵ.
ϵ-*cerradura*(T)	Conjunto de estados del AFN a los que se puede llegar desde cierto estado s del AFN en el conjunto T, sólo en las transiciones ϵ; $= \cup_{s \text{ en } T}$ ϵ-*cerradura*(s).
mover(T, a)	Conjunto de estados del AFN para los cuales hay una transición sobre el símbolo de entrada a, a partir de cierto estado s en T.

Ilustración 1 - Operaciones sobre los estados del AFN.

Tras conocer las operaciones que nos ayudarán a ordenar los estados del AFN, se procede a aplicar el algoritmo de la Ilustración 2. Que nos arroja como resultado las transiciones del AFD resultante.

```
while ( hay un estado sin marcar T en Destados ) {
        marcar T;
        for ( cada símbolo de entrada a ) {
                U = ε-cerradura(mover(T, a));
                if ( U no está en Destados )
                        agregar U como estado sin marcar a Destados;
                Dtran[T, a] = U;
        }
}
```

Ilustración 2 - Algoritmo de Construcción de Subconjuntos.

Objetivo

Programar el algoritmo de la construcción de subconjuntos para realizar la conversión de un Autómata Finito No Determinista a un Autómata Finito Determinista en lenguaje Python.

Desarrollo

En esta sección se abarcará el desarrollo de la práctica.

Metodología

Para esta práctica se optó por utilizar una metodología incremental (Ilustración 3). Ya que esta permite agregar nuevas funcionalidades por medio de distintas fases (llamadas incrementos).

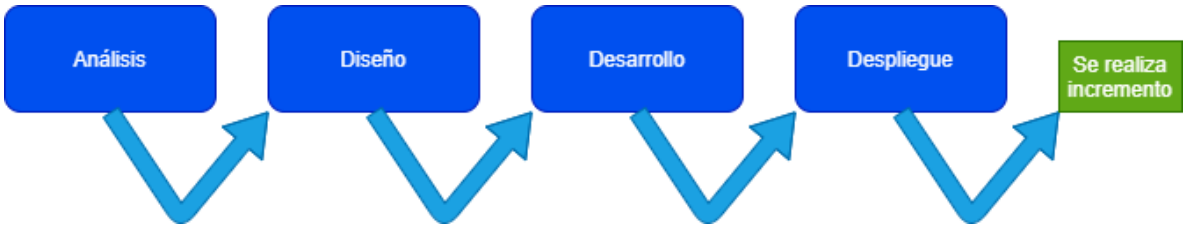

Ilustración 3 - Metodología Incremental.

Estas mismas fases pueden ser reutilizadas en otras prácticas futuras (en este caso los métodos a desarrollar). Esta etapa consiste en:

- Análisis: En esta fase se estudió el algoritmo y su funcionamiento paso por paso y herramienta por herramienta. Se toma como referencia la introducción

de este documento. Se revisó el funcionamiento de las operaciones sobre los estados del AFN.

- Diseño: Consiste en el mapeo de clases, métodos y variables a utilizar en la práctica. Para este caso, sólo se va a desarrollar una clase, la clase ***subsets,*** la cual contiene las variables y métodos indicados en la ilustración 4.

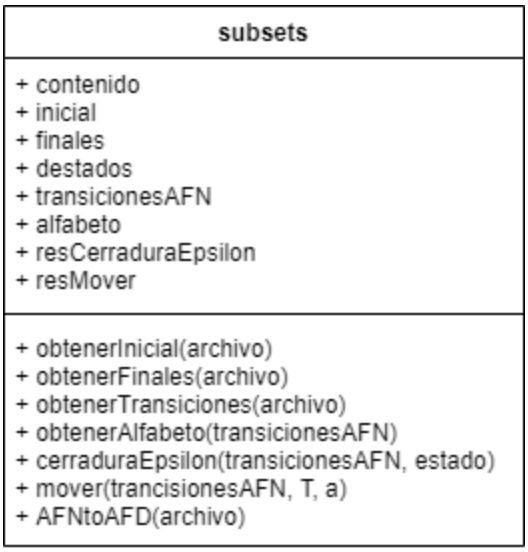

Ilustración 4 - Clase subsets

- Desarrollo: Dentro de esta fase se codifican las partes señaladas en la etapa anterior. Dentro de esta se realizan las pruebas y posibles ajustes para un mejor funcionamiento y optimización. El programa se realizó utilizando el lenguaje Python en su versión 3. El código puede verse en la carpeta ***Código*** anexada en la entrega de esta práctica.
- Despliegue: Se realiza la entrega del incremento realizado con posibilidad de hacer ajustes futuros.

Conclusiones

La práctica realizada cumple con muchas de las características que tiene el algoritmo de construcción de subconjuntos. Me fue más fácil partir de algo ya establecido a manera de pseudo-código, que es como viene en el libro. Sólo faltó realizar la salida del programa, la cual no se pudo completar por falta de tiempo. Pero gracias a las facilidades que me brinda el lenguaje Python, me fue más fácil comprender cómo funciona este algoritmo.

Referencias

[1] A. Aho, M. Lam, R. Sethi y J. Ullman, Compiladores, Principios, Técnicas y Herramientas, Ciudad de México: Pearson Educación, 2008.

[2] J. Hopcroft, R. Motwani y J. Ullman, Introducción a la teoría de autómatas, lenguajes y computación, Madrid: Pearson Educación, 2007.